A DISCRETE $\theta(D)$ ALGORITHM DESIGNED TO INCREASE THE CONVERGENCE SPEED IN SOLVING TRANSPORT-TYPE PROBLEMS

Marius Marinel STĂNESCU[1], Petre STAVRE[2], Dumitru BOLCU[3], Sabin RIZESCU[4], Marcela URSACHE[5]

In this article, we propose a discrete algorithm that can be used to solve transport-type problems. The real improvement brought by applying this original algorithm consists in obtaining a completely filtered and kind of optimal program, following a small number of steps, comparing with other existing algorithms. The outcome will be that the computer memory and the implementation costs will be reduced. The solution is based upon combinatorial methods, and it addresses a certain transport capacity X *that is bounded both above and below* $(d \le X \le D)$. *As a concrete example, we also present a* $(n, m) = (3, 3)$ *numerical application corresponding to a situation of* 3 *suppliers and* 3 *of their beneficiaries.*

Keywords: linear programming, discrete algorithm, transport-type problem.
MSC 2000: 90B20.

1. Introduction

A transport-type problem of classical kind (see [1-2]) usually supposes that the necessary transport capacities are satisfied, whatever the initial transportation planning-program is (unlimited capacities). In fact, the most comprehensive transport-related issues are those with pre-determined transport capacities. The mathematical model is classical. Further, we add the inequalities:

$$d_{ia} \le x_{ia} \le S_{ia}, \tag{1}$$

where x_{ia} represents the transport capacity from the supplier F_i to the beneficiary B_a, $a = \overline{1, m}$; $i = \overline{1, n}$; this capacity cannot be less than d_{ia} (because the transport must be economically justified), and it cannot be greater than S_{ia} (the maximum transport capacity).

[1]Assoc.Prof., Department of Applied Mathematics, University of Craiova, Romania, e-mail: mamas1967@gmail.com
[2]Prof., Department of Applied Mathematics, University of Craiova, Romania
[3]Prof., Faculty of Mechanics, University of Craiova, Romania, e-mail: dbolcu@yahoo.com
[4]Assoc.Prof., Faculty of Mechanics, University of Craiova, Romania, e-mail: sabin_rizescu@yahoo.com
[5]Prof., Faculty of Physics, University of Craiova, Romania, e-mail: ursachemarcela@yahoo.com

This is a bilateral kind of linear programming problem having both its boundaries (the lower and the upper limits) precisely specified.

The inequalities (1) can be brought to the form:

$$0 \le u_{ia} \le D_{ia}, u_{ia} = x_{ia} - d_{ia}, D_{ia} = S_{ia} - d_{ia}. \quad (2)$$

Such a transport-type problem is usually solved out, either by using the (D) algorithm (see [3]) or by adapting the Dantzig-Wolfe decomposition algorithm for this case (see for instance [4]). Moreover, in some other kind of programming models, d and S can be negative (but $d < S$ and thus $S - d > 0$).

As a remark, if n is "big" then $n \cdot m$ is also "big" (and it becomes even bigger when m is "big", too). Even for values of n, m, conveniently chosen, the value of $n \cdot m$ might be too big to be registered in the computing memory. This last case can be solved by keeping an (m,n) appropriate matrix in the memory of the computing device.

Some examples where the Dantzig-Wolfe decomposition algorithm was adapted for classical transport-type problems can be seen in [5] and [6]. Both of them approach the case of two close indeed positioned cranes. These two cranes have to work together by adapting sequences in order to stack freight containers in a certain harbor. Those sequences require that each crane must not stop because of any action of the other one and both of them have to work at their maximum working capacity.

In [7], a decomposition procedure for solving a class of transport-type problems with a linear fractional objective function is discussed. The technique provides not only a solution for the primal problem, but also for the dual problem.

The work paper [8] presents an algorithm designed to minimize the necessary amount of investing expenses designed to increase the working capacity of a transportation network with many inputs and outputs (complex network). In [8], it is also given an example of a transportation network for water or gas distribution.

All these papers contain algorithms that require performing a large number of steps in order to obtain the completely filtered and optimal program. This fact requires the achievement of some laborious calculus and also, a large computer memory.

In order to eliminate the inconveniences related to linear programming with bounded variables, we have developed the following discrete algorithm. This new algorithm can be applied to all classical transport-type problems (technical and/or economical problems, like those arising from large domains such as constructions, design and so on) that fit mathematically (in terms of modelling) to such class of problems.

Classical algorithms used to solve the transport-type problems are based on concepts like base programs filtering, inserting some compensation variables, and so on (see for example [12]). All these elements are added to problem restrictions and lead to a large number of unknown parameteres that require an increase in the computing

memory and a higher time to solve the problem. The proposed algorithm has the advantage of avoiding these difficulties.

2. The discrete algorithm

We start by pointing out that the proposed algorithm is a general type one. It could be applied in any practical model related to whatever transport-type problem is, with no limitations.

In what follows, we consider a transport-type problem having the restrictions $0 \le x_{ia} \le D_{ia}$. It is necessary to determine an initially $\underset{0}{X}$ completely filtered program. This program can be directly obtained or we can obtain a basis program of the system just with the restrictions $X \ge 0$ and after that, if this program is not filtered, we can filter it (see for instance [3]).

In the case we are considering, we adjust the method of minimal cost. Thus, we will have:

a) We consider $D_i = \sum_{a=1}^{n} x_{ia} \left(i = \overline{1, m}\right)$ and $N_a = \sum_{i=1}^{m} x_{ia} \left(a = \overline{1, n}\right)$. When the transport cost, denoted by c_{ia}, has to be minimal (compared to the other transport costs), we shall allocate a merchandise quantity like this

$$x_{ia} = \min\left\{D_i, N_a, D_{ia}\right\}. \tag{3}$$

b) We modify D_i and respectively N_a and we return to a), with another c and the new D, N (c is the smallest among those remaining, without c_{ia} of a).

Thus, we obtain a completely filtered program that can have at most $(m+n-1)$ positive components, but $n \cdot m - (m+n-1)$ components have the form:

$$\underset{0}{x_{jb}} = 0; (j, b) \in K_1^{(0)} \text{ or } \underset{0}{x_{kd}} = D_{kd}; (k, d) \in K_2^{(0)}. \tag{4}$$

c) We arrange $(m+n-1)$ as basic values, denoted by $\left\{\underset{0}{x_{ia}}\right\}, (i, a) \in I^{(0)}$; first, among them, there will be extracted $\underset{0}{x_{ia}} < D_{ia}$ and, next, we will choose the rest of them up to $(m+n-1)$ values, from $\underset{0}{x_{sc}} = D_{sc}$, in ascending values of costs c_{sc}. The other values $\underset{0}{x_{rl}} = D_{rl}$ are considered secondary, $(r, l) \in K_2^{(0)}$. With this completely filtered program $\underset{0}{X}$ and with the established $K_1^{(0)}, K_2^{(0)}$, we may start the algorithm.

Having at our disposal a non-degenerated base program, we want to obtain an algorithm that, using a small number of "steps", leads us to a base program of optimal kind (compared with the classical methods of solving the transport-type problems). This fact is favorable, because the computer memory used in solving such kind of problems is reduced.

Let f be the associated function of the transport-type problem. For a good understanding, we shall define the algorithm that makes possible the passing from the initial step to the next step.

0) We have $\underset{0}{X}, f_0, I^{(0)} = \left\{(i,a) \middle| \underset{0}{x_{ia}} \in \underset{0}{X}\right\}, K_1^{(0)}, K_2^{(0)}$.

1) a) We write $C_1^{(0)} = \left\{c_{ia} \middle| (ia) \in I^{(0)}\right\}$

b) We calculate $c_1^{(0)} = \max\limits_{(ia)\in I^{(0)}} \{c_{ia}\} = c_{jb}$ and we write:

$$\bar{K}_1^{(0)} = \left\{(jd),(ib), d \neq b, i \neq j; \underset{0}{x_{jd}} = 0, \underset{0}{x_{ib}} = 0\right\} \subset K_1^{(0)},$$

$$\bar{K}_2^{(0)} = \left\{(jd),(ib), d \neq b, i \neq j; \underset{0}{x_{jd}} = D_{jd}, \underset{0}{x_{ib}} = D_{ib}\right\} \subset K_2^{(0)}.$$

c) We write $\underset{1}{x_{jb}} = \underset{0}{x_{jb}^{\sigma}}$ $(\sigma = \pm 1)$ and we pass to the point 2).

2) a_1) We calculate $\min\limits_{(jd)(ib)\in\bar{K}_1^{(0)}} \{c_{jd}, c_{ib}\} = \bar{c}^{(0)}$ (c_{jc} or c_{rb}).

a_2) We calculate $\max\limits_{(jd)(ib)\in\bar{K}_2^{(0)}} \{c_{jd}, c_{ib}\} = \overset{=}{c}^{(0)}$ (c_{jl} or c_{kb}).

a_3) We calculate $\min\left\{\bar{c}^{(0)} - c_1^{(0)}, -\left(\overset{=}{c}^{(0)} - c_1^{(0)}\right)\right\} = V^{(0)}$.

If $V^{(0)} = \bar{c}^{(0)} - c_1^{(0)}$, then we pass to the point (b_1).

If $V^{(0)} = -\left(\overset{=}{c}^{(0)} - c_1^{(0)}\right)$, then we pass to the point (b_2).

(b_1) We write $\underset{1}{x} = \theta^+$ instead of $\bar{c}^{(0)}$ in $\underset{0}{X}$ and we pass to (c_1).

(b_2) We write $\underset{1}{x} = \underset{0}{x} - \theta$ instead of $\overset{=}{c}^{(0)}$ in $\underset{0}{X}$ and we pass to (c_2), where $\underset{0}{x}$ is the upper border D.

(c_1) We have (b_1). We admit that $\overset{-(0)}{c} = c_{jc}$ and, thus, $\underset{1}{x_{jc}} = \theta^+$. We seek for a cycle starting with $\theta^+ \to \underset{1}{x_{jb}} = \underset{0}{x_{jb}^-}$ $(\sigma = -)$ so that we have

(1) $\Delta f_0 = \sum c^+ - \sum c^-$ where c^+ are the values (c) of the cycle corresponding to $\underset{0}{x^+}$ and c^- are the values (c) of the cycle corresponding to $\underset{0}{x^-}$.

(c_{11}) If $\Delta f_0 \le 0$ then we may estimate:

(2) $r^{(0)} = \min\left\{ \underset{0}{x_{ia}} \,\middle|\, included \;\; as \; x_{ia}^- \; in\, to \; the \; cycle \right\}$;

(3) $R^{(0)} = \min\left\{ D_{ia} - \underset{0}{x_{ia}} \,\middle|\, if \;\; \underset{0}{x_{ia}} \;\; appears \, in\, to \; the \; cycle \; as \; x_{ia}^+ \right\}$;

(4) $\theta = \min\left\{ D_{jc}, r^{(0)}, R^{(0)} \right\}$

With θ chosen in this way, we have obtained a new completely filtered program $\underset{1}{X}$ and $f_1 = f_0 + \Delta f_0 \le f_0$. With $\underset{1}{X}$, we resume the algorithm $(p := p+1;\; p \ge 0)$. We have removed c_{jb}.

Remark 1. At this moment, in order to increase the convergence of the algorithm (by reducing the number of steps to get the optimally completed filtered program), we shall search all cycles that begin with $\theta^+ \to \underset{0}{x_{jb}^-}$ also having $\Delta f_0 \le 0$ and we will choose the one where $\Delta f_0 \le 0$ has the smallest values.

(c_{12}) If there is not a cycle with $\Delta f_0 \le 0$, then we make $\overline{K}_1^{(0)}, \overline{K}_{1act}^{(0)} = \overline{K}_1^{(0)} - \{c_{jc}\}$ and we resume the algorithm to $\underset{0}{X}, C_1^{(0)}, \overline{K}_{1act}^{(0)}, \overline{K}_2^{(0)}$.

(c_2) We have $\underset{1}{x} = \underset{0}{x} - \theta$. Let be $\overset{=(0)}{c} = c_{jc}, (j,c) \in \overline{K}_2^{(0)}$. We write $\underset{1}{x_{jc}} = D_{jc} - \theta$ (because $\underset{1}{x_{jc}} = D_{jc}$) and we note $\underset{1}{x_{jc}} = D_{jc}^-$.

(c_{21}) We build the cycles that start with $D_{jc}^- \to \underset{0}{x_{jb}^+}$ $(\sigma = +)$ having $\Delta f_0 \le 0$, each of them. If these exist, we choose θ, then through (4) we shall obtain a new

program $\underset{1}{X}$ with $f_1 = f_0 + \theta \Delta f_0 \le f_0$. We remove c_{jb} and we resume the algorithm.

(c_{22}) If a cycle with $\Delta f_0 \le 0$ does not exist, then we bring up to date $\overline{K}_2^{(0)}$, $\overline{K}_{2act}^{(0)} = \overline{K}_2^{(0)} - \{c_{jc}\}$. With the new data $\underset{1}{X}, C_1^{(0)}, \overline{K}_1^{(0)}, \overline{K}_{2act}^{(0)}$, we resume the algorithm.

3) If after the updates made at $(c_{11}), (c_{22})$ we obtain $\overline{K}_{1act...act}^{(0)} = \Phi, \overline{K}_{2act...act}^{(0)} = \Phi$, then we bring $C_1^{(0)}$ up-to-date, or specifically: $C_{1act}^{(0)} = C_1^{(0)} - \{c_{jb}\}$.

With the new brought-to-date $\underset{0}{X}, C_{1act}^{(0)}$, the algorithm will be resumed.

Proposition If all values $x_{ia} \in \underset{0}{X}$ are fixed, namely $C_{1act...act}^{(0)} = \Phi$, then STOP $\underset{0}{X}$ is a completely filtered and optimal program (or for $p > 0$, if $C_{1act...act}^{(0)} = \Phi$, then $\underset{0}{X}$ is optimal).

Proof. No matter what the case $b_1)$ or $b_2)$ is, the equality $f_1 = f_0 + \Delta f_0$ must remain in place and if $\Delta f_0 < 0$, then $f_1 < f_0$, meaning that $\sum c^+ - \sum c^- < 0$.

In the case $b_1)$, a cycle starts with $\theta^+ \to \underset{1}{x}_{jb} = \underset{0}{x}_{jb} - \theta \quad (\theta > 0)$ and in that cycle, we shall have a vertex fulfilling $\underset{0}{x}_{rs} - \theta$ as well as $\underset{0}{x}_{rs} + \theta$ (noted by $\underset{0}{x}^-$ and $\underset{0}{x}^+$ respectively).

For $\left\{\underset{0}{x}_{rs} - \theta\right\}$ we insert the condition $\underset{0}{x}_{rs} - \theta \ge 0$ or $\underset{0}{x}_{rs} \ge \theta$, or being even more specific: $\theta \le \min\left\{\underset{0}{x}_{rs}\right\} \overset{def}{=} r^{(0)}$.

For $\left\{\underset{0}{x}_{rs} + \theta\right\}$ we insert the condition $\underset{0}{x}_{rs} + \theta \le D_{rs}$, $(\forall)(r, s)$, and $\theta \le D_{rs} - \underset{0}{x}_{rs}$, namely: $\theta \le \min\left\{D_{rs} - \underset{0}{x}_{rs}\right\} \overset{def}{=} R^{(0)}$.

But $\theta = \underset{1}{x}_{jc}$ (or $\theta = \underset{1}{x}_{rb}$) so the conditions (2), (3), (4) are fulfilled.

If $\theta = 0$, then we make $\overline{K}_1^{(0)}, \overline{K}_{1act}^{(0)} = \overline{K}_1^{(0)} - \{c_{jc}\}$ and we resume the algorithm.

We shall proceed in the same way as we did in the cases b_2) and c_2).

Because $\underset{1}{x}_{jc} = D_{jc} - \theta$, it results that $\underset{1}{x}_{jb} = \underset{0}{x}_{jb} + \theta$, and in the ongoing cycle, we will only have the terms $\underset{0}{x}_{rs} - \theta, \underset{0}{x}_{rs} + \theta$. If we put the conditions: $\underset{1}{x}_{jc} \geq 0, \underset{0}{x}_{rs} - \theta \geq 0, \underset{0}{x}_{jb} + \theta \leq D_{jb}, \underset{0}{x}_{rs} + \theta \leq D_{rs}$, then the relations (2), (3), (4) will result again.

3. Numerical Applications

Remark 2. In order to understand better how the algorithm steps are checked, and not to load the paper with a large volume of calculus, we shall consider a simple numerical application.

In addition, choosing a different number of suppliers and beneficiaries is not relevant for transport-type problems. Anyway, our discrete algorithm presents a general situation in which n and m can take any integer values.

To justify the choice of a "small" example as a numerical application and somehow, to highlight a certain advantage of the proposed algorithm (small number of variables and reduced working time) compared with classical algorithms, we have to point out that the number of variables in the case we will choose (for $m = n = 3$) is actually equal to $m \cdot n$, meaning 9 for our algorithm. This is quite an improvement comparing to the classical case where, to this number, the compensation variables $m + n$, meaning 6, are added, resulting in a total number of 15 variables. If *m* and *n* increase, the number of variables in the classical variant, increase almost exponentially (for example, if we consider the parameters $m = 3$ and $n = 5$, for the proposed algorithm, we have to deal with 15 variables which is considerably better than for the classical situation where we have to deal with 23 variables).

Problems of such kind might indeed be balanced or not. When the problem is not balanced, it can always be solved as a balanced one by imaginary adding a supplier or, depending on the case, a beneficiary, in order to take over the remaining amount of „supply" that will never actually be „supplied" or „demand" that, in fact, will never be „demanded".

In the following table (T), we give the data of a transport-type problem (balanced), with the restrictions $0 \leq x_{ia} \leq D_{ia}$. We have considered the situation of

three suppliers F_i and three beneficiaries B_a $\left(i, a = \overline{1,3}\right)$. Each and every cell of the table contains the ratio $\left(\frac{c_{ia}}{D_{ia}}\right)$, the numbers c_{ia} meaning the unitary transport costs.

(T)

	B_1	B_2	B_3	Disposable
F_1	$\frac{2}{3}$	$\frac{5}{13}$	$\frac{4}{1}$	15
F_2	$\frac{4}{10}$	$\frac{7}{2}$	$\frac{9}{10}$	10
F_3	$\frac{4}{9}$	$\frac{8}{15}$	$\frac{1}{12}$	20
Necessary	20	13	12	45

$\Rightarrow$

3		12	
10^-	$\rightarrow$	θ^+	
$\uparrow$		$\downarrow$	
7	$\leftarrow$	1^-	12

$\Rightarrow \underset{0}{X}$, $f_0 = 154$

Indeed, we have obtained a completely filtered program $\underset{0}{X}$, because there are $(m+n-1)$ components $0 \le x_{ia} \le D_{ia}$, and the other secondary values are 0 or D. We can write:

$0^{(0)})$ $\underset{0}{X}, f_0 = 154, I^{(0)} = \{(21), (12), (31), (32), (33)\}, K_1^{(0)} = \{(13), (22), (23)\}, K_2^{(0)} = \{(11)\}.$

Obviously, we can organize $K_2^{(0)}$ by any couple (ia) that satisfies the equality: $\underset{0}{x_{ia}} = D_{ia}$. For $(ia) = (21)$ and for $(ia) = (33)$ as well, we will show how the algorithm works:

$1^{(0)})$ *a*) We write $C_1^{(0)}(D) = \{c_{11} = 2, c_{12} = 5, c_{31} = 4, c_{32} = 8, c_{33} = 1\}$.

b) We choose $C_0^{(1)} = 8 = c_{32}$. It results that $\overline{K}_1^{(0)} = \{(22)\}, \overline{K}_2^{(0)} = \Phi$.

c) We write $\underset{1}{x_{32}} = \underset{0}{x_{32}^{\sigma}} = 1^{\sigma}$ $(\sigma = \pm 1)$.

$2^{(0)})$ a_1) We calculate $\min\limits_{(3a)(i2) \in \overline{K}_1^{(0)}} \{c_{3a}, c_{i2}\} = 7 = c_{22}$.

a_2) We calculate $\max\limits_{(3a)(i2) \in \overline{K}_1^{(0)}} \{c_{3a}, c_{i2}\}$. Because on the line 3 and column 2, a secondary value beyond the upper border $\left(\overline{K}_2^{(0)} = \Phi\right)$ does not exist, we choose:

c_1) $x_{22}^{1} = \theta^+$, $x_{32}^{1} = 1^-$ that generate a cycle, so we have:

$$\Delta f_0 = \theta(7 - 8 + 4 - 4) = -1 < 0.$$

We calculate: $r^{(0)} = \min\{1, 10\} = 1$, $R^{(0)} = \min\{2, 7\} = 2$ and $\theta = \min\{2, 1, 2\} = 1$.

A new completely filtered program X_1 results with $f_1 = f_0 + 1 \cdot \Delta f_0 = 153$, where:

X_1

3	12	
9	1	
8		12

$f_1 = 153$

With X_1, we resume the algorithm

$0^{(1)}$) $X_1, f_1, I^{(1)} = \{(21), (31), (12), (22), (33)\}$, $K_1^{(1)} = \{(13), (23)\}$, $K_2^{(1)} = \{(11)\}$

(and here $K_2^{(1)}$ can be $K_2^{(1)} = \{(33)\}$.

$1^{(1)}$) *a*) We write $C_1^{(1)} = \{c_{21} = 4, c_{31} = 4, c_{12} = 5, c_{22} = 7, c_{33} = 1\}$.

b) $C_1^{(1)} = c_{22} = 7$. It results that $\overline{K}_1^{(1)} = \{(23)\}$, $\overline{K}_2^{(1)} = \Phi$.

c) We write $x_{22}^{2} = x_{22}^{\sigma}{}_{1} = 1^{\sigma}$ $(\sigma = \pm 1)$.

$2^{(1)}$) a_1) We calculate $\min\limits_{(2a)(i2) \in \overline{K}_1^{(1)}} \{c_{2a}, c_{i2}\} = 9 = c_{23}$.

a_2) $\max\limits_{(2a)(i2) \in \overline{K}_2^{(1)}} \{c_{2a}, c_{i2}\} = \Phi$.

It results that $\sigma = -$; $x_{22}^{2} = 1^-$; $x_{23} = \theta^+$.

Because $\theta^+ \to 1^-$ does not accomplish a complete algorithm and on the line of $x_{22} = 1$, respectively on its column do not exist (c) secondary values different from $c_{23} = 9$, we may write: $\overline{K}_{1act}^{(1)} = \overline{K}_1^{(1)} - \{c_{22}\} = \Phi$. Because $\overline{K}_{1act}^{(1)} = \Phi$, $\overline{K}_2^{(1)} = \Phi$, we bring up-to-date $C_1^{(1)}$, or being even more specifically: $C_{1act}^{(1)} = C_1^{(1)} - \{c_{22}\} = \{c_{11} = 4, c_{12} = 5, c_{31} = 4, c_{33} = 1\}$.

$1_{act}^{(1)}$) *a*) $X_1, C_{1act}^{(1)}$;

b) $C^{(1)}_{1act} = 5 = c_{12} \Rightarrow \overline{K}^{(1)}_1 = \{(13)\}, \overline{K}^{(1)}_2 = \{(11)\}.$

c) We write $\underset{2}{x_{12}} = 12^{\sigma}$ $(\sigma = \pm 1).$

$2^{(1)}$) a_1) $\min\limits_{(1a)(i2)\in\overline{K}^{(1)}_1} \{c_{1a}, c_{i2}\} = 4 = c_{13}.$

a_2) $\max\limits_{(1a)(i2)\in\overline{K}^{(1)}_2} \{c_{1a}, c_{i2}\} = c_{11} = 2$ (because $(11) \in \overline{K}^{(1)}_2$).

a_3) $\min\{4-5=-1, -(2-5)=3\} = -1 = c_{13} - c_{12} = V^{(1)}$. We step to b_1).

b_1) We write $\underset{2}{x_{13}} = \theta^+ \Rightarrow c_1$) $\underset{2}{x_{12}} = 12^-$ $(\sigma = -).$

But $\theta^+ \to 12^-$ does not achieve a cycle with $\Delta f_1 \le 0$. We will fix this.

We bring up-to-date $\overline{K}^{(1)}_{1act}$ or, more specific: $\overline{K}^{(1)}_{1act.act} = \overline{K}^{(1)}_{1act} - \{c_{13}\} = \Phi$.

Because $\overline{K}^{(1)}_2 = \{(11)\}$ has an element on the line of $x_{12} = 12$, we may continue:

$1^{(1)}_{act.act}$) *a*) $\underset{1}{X}, C^{(1)}_{1act}, \overline{K}^{(1)}_{1act.act} = \Phi, \overline{K}^{(1)}_2 = \{(11)\}.$

b) $C^{(1)}_{1act} = 5 = c_{12}$. We write $\underset{2}{x_{12}} = 12^+.$

$2^{(1)}_{act.}$) Because $\overline{K}^{(1)}_{1act.act} = \Phi$ we may step right to a_2) and it results that $c_{11} = 2$.

b_2) We write $\underset{2}{x_{11}} = \underset{1}{x^-_{11}} = 3^-;\ \underset{2}{x_{12}} = \underset{1}{x^+_{12}} = 12^+.$

We have formed a cycle:

$$\begin{array}{ccc} 3^- & \to & 12^+ \\ \uparrow & & \downarrow \\ 9^+ & \leftarrow & 1^- \end{array} \qquad \Delta f_1 = \theta(5-2+4-7) = 0.$$

We calculate:

$\min\{3,1\} = 1 = r^{(1)}; \min\{13-9, 13-12\} = 1 = R^{(1)};\ \theta = \min\{D_{11} = 3, r^{(1)}, R^{(1)}\} = 1.$

It results a new completely filtered program $\underset{2}{X}$ with $f_2 = f_1 = 153$.

X_2

2	13	
10		
8		12

and we resume the algorithm. (We have considered c_{22} as fixed, because $\theta = x_{22} = 1$). We have eliminated $c_{jb} = c_{12}$.

We have X_2, $I^{(2)}$, $K_1^{(2)} \neq \Phi$, $K_2^{(2)} = \Phi$, because only $(m-n-1)=5$ basic values do exist (X_2 is already a filtered base program).

We resume the algorithm and we immediately obtain that $C^{(2)}_{1act.act.act} = \Phi$ STOP, X_2 is kind of optimal and identical with X_1, because $f_1 = f_2 = f_{\min} = 153$.

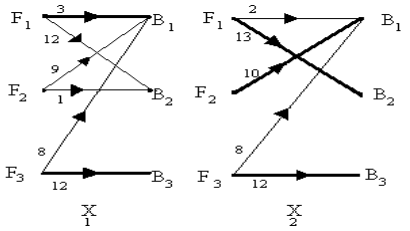

Fig. 1. The optimal allocation graphs X_1, X_2.

4. Conclusions

In X_1, (F_1B_1), (F_3B_3) are saturated (at maximum capacity) and in X_2, (F_1B_2), (F_2B_1), (F_3B_3) are saturated (at maximum capacity). From an economical point of view (see for instance [9-16]) we will choose the path allocation that fits the best, considering the spreading of transport capacities.

More else, resuming a linear convex connection between X_1 and X_2, we can find more other optimal allocations (diversifying the optimal allocations). We will retain the only value that does fit with different more other requests (spreading of capacities).

$$X = a\,X_1 + (1-a)\,X_2 = \begin{pmatrix} 2+a & 13-a & 0 \\ 10-a & a & 0 \\ 8 & 0 & 12 \end{pmatrix}, \quad a \in [0,1].$$

If we want only integer-numbers solutions, the only optimal programs will be X_1 and X_2.

Remark 3. If after $\underset{2}{X}$ is obtained, we take into account that $c_{23} = 9, c_{13} = 4$ will be eliminated, then it will directly result that $C^{(2)}_{1act.act.act} = \Phi$, so any other step will be simply unnecessary to be done.

Acknowledgement

This work was supported by CNCSIS –UEFISCSU, project number PNII – IDEI 289/2008 and ANCS project Development of biomimetic design methodology with reverse engineering in cognitive recognition and control of biomimetic robot/Reverse Engineering in Cognitive Recognition And Control Of Biomimetics Structures -PNII – CAPACITATI - Bilateral Projects - 2010

R E F E R E N C E S

[1] *L.R. Ford, D.R. Fulkerson,* "Flows in Networks", Princeton University Press. Princeton N.Y. 1962.

[2] *I. Newton,* "Philosophiae Naturalis Principia Mathematica (second edition)", translated in romanian language by Romanian Academy Publishers, 1956.

[3] *P. Stavre,* "Linear Programming", Universitaria Publishers, Craiova, 1998.

[4] *G.B. Dantzig, P. Wolfe,* "Linear Programming and Extension", Princeton Univ. Press, 1957. N.J 1963.

[5] *R.F. Dell, J.O. Royset, I. Zyngiridis,* "Optimizing container movements using one and two automated stacking cranes", Journal of Industrial and Management Optimization, **vol.** 5, issue: 2, 2009, pp. 285-302.

[6] *A. Azaron, O. Tang, R. Tavakkoli-Moghaddam,* "Dynamic lot sizing problem with continuous-time Markovian production cost", International Journal of Production Economics, **vol.** 120, issue 2, 2009, pp. 607-612.

[7] *R. Gupta,* "Decomposition Method and Transportation Type Problems with a Fractional Objective Function", ZAMM - Journal of Applied Mathematics and Mechanics, **vol.** 57, issue 2, 1977, pp. 81-88.

[8] *D. Dusmanescu, D. Enachescu, M. Matei,* "Optimal investment level for increase the capacity of an transportation network", Innovation and Knowledge Management in Twin Track Economies: Challenges & Solutions, **vol.** 1-3, 2009, pp. 1158-1163.

[9] *M. Dragomirescu, M. Maliţa,* "Quadratic Programming", Scientific Publishers, Bucharest, 1968.

[10] *K. Lancaster,* "Mathematics economically analysis", Scientific Publishers, Bucharest, 1970.

[11] *H. Williams, P. Chichester,* "Modern Mathematical Method of Optimization", Academic Verlag, Berlin 1993.

[12] *P. Stavre,* "Applied Mathematics in Economy", Scrisul Romanesc Publishers, Craiova, 1982.

[13] *M.M. Stănescu, D. Bolcu, I. Ciucă,* "Operational Research. Programming in integer numbers", Universitaria Publishers, Craiova, 2010.

[14] *P. Stavre, M.M. Stănescu,* "On Benders and S(P<); (P>) algorithms", Annals of University of Craiova (Series Mechanics), no. 1, 2003, pp. 219-232.

[15] *O. Dogaru, M. Postolache, M. Constantinescu,* "Optimality conditions for a family of curves", U.P.B. Sci. Bull., Series A, **vol.** 73, issue 3, 2011, pp. 3-12.

[16] *I. Lungu, A. Pîrjan, D.M. Petroşanu,* "Optimizing the computation of eigenvalues using graphics processing units", U.P.B. Sci. Bull., Series A, **vol.** 74, issue 3, 2012, pp. 21-36.